UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Final Paper 2019

Title: CARPARK – A Low-Cost, Implementable Smart Parking Solution for University Campuses

Author: Thabo Kopane

Project Abbreviation: CARPARK

Supervisor(s): A/Prof. Michelle Kuttel

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	
Theoretical Analysis	*0*	*25*	
Experiment Design and Execution	*0*	*20*	
System Development and Implementation	*0*	*20*	
Results, Findings and Conclusion	*10*	*20*	
Aim Formulation and Background Work	*10*	*15*	
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	
Total marks		**80**	

CARPARK – A Low-Cost, Implementable Smart Parking Solution for University Campuses*

Thabo Kopane
Department of Computer Science
University of Cape Town
Rondebosch, Cape Town, South Africa
Kpntha001@myuct.ac.za

ABSTRACT

Parking at the University of Cape Town is time consuming and stressful for many students and staff members. The problem is that there are more cars that park on campus than available parking spaces, the university is not planning on expanding parking spaces in the immediate future.

We presented a carparking system; a mobile application that works with an Arduino-based sensor. The sensor scans cars going in and out of carparks and uploads the information to an API. The application downloads the data from the API and displays the information to end users. The focus of the paper is on the design of the mobile application following the user centred design methodology. The paper will be investigating how the design of a mobile application influences how it is used.

We conducted interviews for the paper and interactive prototype assessments, the app prototype assessments were conducted in a focus group.

Our assessments showed that parking affected a lot of users; users prefer minimalist mobile applications and applications that are focused on either one thing or things in the same domain, users are open to changing how they drive or park on campus if that means they can reduce the amount of time spent looking for parking.

CCS CONCEPTS

• Human Centred Computing • HCI • Collaborative and social computing, ubiquitous and mobile computing

KEYWORDS

Mobile application; user centred design; Arduino; smart parking; crowdsourcing;

1. INTRODUCTION

There are more cars on the road today, there are more people who drive to campus than there were a few years ago and the university admits more students than it had ever before; this means there is an increase in demand for parking on campus.

The university has more facilities than before; there are library, computer and other important facilities that keep students on campuses longer [24].

The increased demand for parking has not been met. The university has said that they are not planning on building more carparks in the immediate future [24]. This has meant that parking at certain campuses during certain times of the day is harder and time consuming because carparks are full. Students are more likely to park illegally or in hazardous areas due to lack of available parking or information about available parking[23]. Khattak and Polak[1993] show that, having access to real-time parking information prior to driving to a shopping centre, town or an office was effective in helping users choose under-utilised parking spaces and also reduced the time users would have spent looking for parking[4].

There are many smart parking solutions available[6], some are designed for small towns, malls or even big cities, however, most parking solutions are not made for university campuses, mostly because of the business models that the systems operate with and the costs of implementing the solutions[6].

User Centred Design (UCD), sometimes known as Human Centred Design (HCD), is a design philosophy whose main guiding principle is the involvement of end users in the design process[27, 28]. The goal of UCD is to learn about users and their needs and the potential object usage in order to design tailored objects[7]. A popular software development methodology is the agile methodology, where systems and/or software products are built iteratively and with clients often involved (this is especially the case when it is an inhouse development team)[36]. Although, agile is not UCD, it has some elements of UCD and the two paradigms can be paired with one another with mixed results[26].

The agile manifesto “customer collaboration over contract negotiations” is similar to UCD’s focus on “form follows function”[15, 36].

UCD champions function over form, the functionality of the design is more important than how it looks; many design paradigms would often focus on form, for UCD, “form follows function”, with user guidance[15].

AIM

We wanted to find out the influence of designs methodologies on mobile apps and how they are used. For this project, we researched ways to give potential users parking information and how that information may be used to reduce parking congestion

by using under-utilised parking spaces and in the future, introduce alternative and effective means such as carpooling. We developed a smart parking solution using the Arduino framework, and designed a mobile app to display that information following UCD. Many smart parking systems in literature that have companion mobile apps, are not designed with the same attention as the “system” itself. Because our focus is on low-cost solutions, we will try to make use of Free and Open Source Software wherever possible.

2. RELATED WORK

2.1 User Centred Design

Designing is more about the users and their experiences now, because it has become important to know how people will use an item, and whether that item adds value into people’s lives[28]. User centred design is a design process in which the designer studies potential users to learn about their needs, wants and tasks[27]. Most software projects involve users during the usability testing of the projects, this is what is usually taught about software engineering in undergraduate courses[40]. Marc Steen et al[2007] define UCD as the active involvement of users for clear understanding of who the users are and how this item will fit in their lives[7]. The paper highlights the advantages of involving users during the design process; some of the advantages include ensuring good product fit and improving product usage. The authors also state that extensive user involvement in projects may stifle innovation, especially innovation of new products because users often base their opinions on products they already know.

UCD emerged as a reaction to the preoccupation of designers with designing for form, and little regard to users. The design ethos was shifting to focus on how objects were used[15]. In the paper Towards User Design[2005] Johan Redström states that the changes to user design did not happen immediately, they were a gradual evolution until the focus was mostly on users[5, 7, 28]. The focus of designers went from form, to function, to communication and finally to user experience. Function is what function the object will have, Communication is how the object communicates its uses to designers and to consumers, and user experience is solely about the user, the user’s needs, wants, experiences and desires.

Users’ considerations used to be a controversial idea in the design sphere according to Redström. One of the changes that were brought about was that, designers realised they could not impose objects on users anymore[18]. In The tall office building artistically considered[15], Sullivan introduced a popular design saying: “form follows function”, most modern designs and design schools follow this process[28].
Redström pushes back against the preoccupation with users, and argues for a middle ground as he says: “There cannot be users of things that do not yet exist”, a tight fit may bring about imitative innovation.

An asocial hiking application is a weird idea, the authors do note that there are limitations to the application can and will be used due to cultural and social considerations[3]. The mobile application was created following UCD, the authors conducted a survey at the onset of the project to find out what people thought of the idea; only 44% of the respondents thought it was a good idea and 40% were confused as to what problem the app is solving. The app received positive reviews during the assessments and contextual inquiries because most of the users who prefer solitary hikes said it was good idea, and they were happy with the design considerations. This project shown the different and effective methods of following UCD, and hence the success in how it was received when finished.

The National Geoparks project’s aim was to enhance user experience and make sure that users are engaged throughout the tours[3]. The apps were going to be integrated into the geopark tours. The designers of the app conducted interviews with the users to have an in-depth idea of who the users were, what were their needs and what they did at the parks. The users wanted to have targeted user modes, provide information about Geopoints (Geopoints are important landmarks) through GPS activation, a short summary when a tour is completed and at the beginning of the tour. The research for this app made extensive use of UCD, and concluded that making a perfect UCD app is close to impossible due to the cost and the different user needs.

UCD and agile development methodologies are not paired often, there is ongoing research about how to pair UCD and agile development in an optimal manner that is also affordable[7, 8]. UCD can slow down the pace of software development and it is costly to implement and that is often cited as the main reason for not pairing the two, the other reason is the difference between a client and a user. Most agile development teams have a client and that client may not be the end-user of the product and the team may not have access to the end user[8].

2.2 Parking applications

Lin et al[2017] completed a survey of different smart parking solution that make use of either IoT, smartphones and sometimes these are parking solutions without an app companion[6]. They went through most of the smart parking solutions presented in literature from 2000-2016, our focus is on the literature after 2010 because that is around the date when smartphones were rising in popularity and ubiquity. Most of the parking systems in the paper that have a mobile app, were not focussing on the interface of the app, but on things such as the GPS, gyroscope and other additional tools in a smartphone that may help in tracking[6].
ParkJam is a crowdsourcing parking application that uses content based algorithms along with user data to present available parking spots to users[12]. A user will state in the app any activity relating to finding a parking spot, a content-based algorithm will then

verify the information before it is available to other users. The users are required for the app to function optimally. The authors only tested the usability at the end of the design[16, 17]and did not involve users during the design of the main features.

Grazioli and Picone et al[2013] created an app that in addition to crowdsourcing, has features such as; wayfinding and starting a parking session[9].The app was designed to either integrate with an already existing parking system, or organisations can build a system around it; it has an option to include a parking controller app if an organisation has one.
The app only mentions conducting usability tests at the end, no users were consulted during the design phase.
Fraifer and Fernström[2017] made a smart parking system following UCD process[10]. The UCD process is applied to the parking app, but they emphasise the importance of UCD for everything in the system, the combination of UCD and IoT is emphasised[11]. The developers assessed the behaviour of carpark users, commissioned a user survey and designed a solution tailored for the users in collaboration with the users. The paper mentions the importance of UCD for Internet of Things (IoT) since IoT devices are being used more and for different things[3, 31]. The main issue with this paper is that they do not go in depth about how UCD was applied for different components, and the mobile app in particular.

Different types of users, wayfinding, parking reservation, crowdsourcing, sensors and cameras are some of the features that are relevant and important for this project, because some of them can work in the context of UCT.
One of the main issues with most of the systems mentioned or in parking literature is the mobile device, the app is usually designed after the "system" and not during the system design[6,13,14]. If a parking system has an app companion, that app may very well be the system for users and the experience with that app should be given a high priority as well.

3. DESIGN

The type of research for this paper is qualitative research; we are going to conduct user assessments to assess the design, interactions, the speed and the overall performance of the application and the system.
We are going to discuss how the research was designed and how it will be assessed, and how the system that backs the research was designed.

3.1 Research Design

In this paper we aim to find out the influence of a design process on the product, the users and the interaction of the product with the users.

The research question being investigated is:

Can a mobile application, designed following the UCD methodology, guide students and staff in using under-utilised carparks?
The research question focuses on the mobile application and the design of the app. In order to spend less time looking for parking, and reduce the stress associated; what should users be doing?

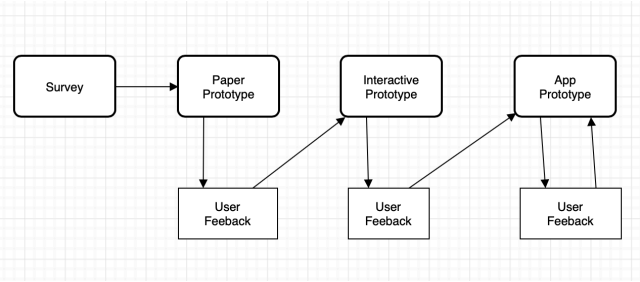


Figure1. Overall research

3.1.1 Survey

Surveys are a quick and affordable process to learn about a group of people or potential users. Surveys are a less intensive method for gathering information about users than something like a questionnaire or interviews [25]. The survey can be a form of requirements gathering and for this project, it was used for that purpose. User centred design requires the involvement of users throughout the design process and therefore it is very important to involve users from the start of the project [7, 28].
The list of questions used during the survey for requirement gathering are attached in the appendix. Potential users were asked about parking apps (if they used parking apps), the problems that they may have experienced parking at the university campuses and the potential solutions to those problems in general and in the form of features for a smart parking application.

3.1.2 Paper Prototype

The paper prototype is an important design phase in the process of designing objects, the designer creates different visual mock-ups of what the product might look like[37, 38]. The paper prototype is very cheap and thus iterating on a paper prototype is faster and easier than all the other prototypes that will be mentioned[37]. The features that were chosen for the paper prototype (mostly from the survey results) were ones that were plausible for the constraints of the project and could be implemented within the project's time frame. There are more features and ideas about the application that users would have wanted to see that did not make it onto the prototype, the most popular amongst them were: being able to reserve parking, an online queueing system and carpooling services. Some of the features were rejected because they are costly to implement and some of them to work require everyone who parks on campus to use the application for parking (the online queueing system is an example) and others require more time to implement.

The paper prototype was evaluated with one-on-one interviews. One-on-one interviews are a detailed way to either learn about subjects or collect user feedback[22]. One of the advantages of one-on-one interviews is that users are free to say anything that they deem relevant for the purposes of the research[23]; and therefore users can often mention things that the researcher may have missed.
An interview script was created for this prototype to find out what users thought of the application and its interactions after the users went through the application, the script is attached to the appendix. The evaluation was conducted with ten users who were recruited through the adverts on campus and using QR (Quick Response code) scanner. Some of the questions that users were asked, which are attached in appendix, include: What problems did you have with the application and which additional features would you like to see in the next update?

3.1.3 Interactive Prototype

The interactive prototype builds onto the paper prototype, and addresses the issues that users found, whilst making the prototype feel like an actual app that users can interact with. The paper prototype was drawn on actual paper and thus lacked important visual cues such as colour, and the interactive prototype looks more realistic and how the final prototype (the app) should look. Proto.io was used to for the interactive prototype, Proto.io is mock-up web application for mobile applications, adds interactions and an option to convert some of the UI to usable code.

We evaluated the interactive prototype with both one-on-one interviews and surveys. We recruited users using the email addresses of students from the survey who stated their interest in usability tests, and we also used the WhatsApp messenger group chat functionality. The one-on-one interviews were assessed with the same questions as the survey, the reason is that we wanted to measure whether there are discrepancies between survey and the interview. We conducted the interactive prototype evaluation with five users for the interview, and five filled in the survey.

3.1.3 App prototype

The application prototype is the actual app launched on devices. The application prototype stage is the stage where the most realistic and important features of the app are tested and should be working, with a few exceptions. The prototype fixes problems experienced in the interactive prototyping by either adding, removing or modifying features.

The app prototype was evaluated with a focus group that included a walkthrough of the app and then a series of questions that users can discuss afterwards.
Focus groups are conducted with a group of people from a minimum of three to as big as ten people at a time. Focus groups can aid in collecting a lot of information in a short space of time with users discussing and interjecting with one another[23, 34]. The discussions in a focus group allow users to bring up features, problems or solutions that might not be possible when the user is by themselves or even a one-on-one interview.
We conducted the focus group with six participants; three male and three female. All participants were students who drove and parked on campus regularly, most were recruited through campus ads and WhatsApp Messenger groups. The users did a walkthrough of the application individually for about 10 minutes and then the host of the focus group asked them questions about the usability of the application, the design and the design cues used and the functions they serve.

The questions for the paper, interactive and the app prototype are similar or identical to one another. We made them similar because we wanted to ensure that the internal consistency of the assessments, what we are testing, are maintained. The users may have been different from all the usability tests (most of the users tested on the interactive prototype were not tested on the paper prototype, similarly with the app prototype.) and these questions ensured that the results and conclusions reached from the studies remain consistent.

4. IMPLEMENTATIONS

4.1 System Design Constraints

The constraints of the mobile application, the environment in which the application would be used and the types of users who would benefit from using the application are as follows:
The application is designed for smartphone use, i.e. not tablet use.
The mobile device should be running either iOS 9 and above or Android 6 and above.
The mobile device should have access to the internet
For a single task, all else being equal, a user is supposed to launch an app and get information about a single carpark in under 1 minute.

These are the current constraints of the application; they may change with additions or modifications to the application.

4.2 System Implementation

The mobile application was developed using the agile development method[8, 36]. The application was developed with C# and the Xamarin development framework, which allows users to develop either for iOS smartphones, Android smartphones or both the iOS and Android smartphones simultaneously using shared code[19]. The reason that C# and Xamarin were chosen for this project is the ability to develop for both Android and iOS on one machine with the same language. There are other cross-platform development frameworks such as React-Native and/or Ionic, that use JavaScript and JavaScript libraries[39]. C# and Xamarin have strong emphasis on object-oriented programming

(OOP) and Model-View Controller (MVC) respectively. C# has a similar syntax to Java, a language that we are familiar with.

The mobile app fetches information from the Rest API and displays that information to users. REST is an API designed to take advantage of existing protocols; there are no additional libraries required, developers can use the API with different types of data, the API can handle multiple calls and change structure quickly and with relative ease[21, 32]**.** The Rest API for this project, was developed using Django framework. Django is an open source web application framework that is widely used if the web application would have been developed using Python as the main programming language[20]. Django follows the MVC pattern where it is known as the Model View Template (MVT). In this project, the Template is not used because it is replaced with a Serializer, a component that converts data structures into formats that could be read or stored. The serializer stores the data as a JSON object, which the app retrieves and deserializes back to data the program can read.
The Arduino sensor scans cars going in and out of a carpark and updates the data on the API. The app retrieves the data from the API and displays the information on parking to the end-users.

The image below shows an overall interaction of the three phases: The Arduino sensor, the REST application and the mobile application.

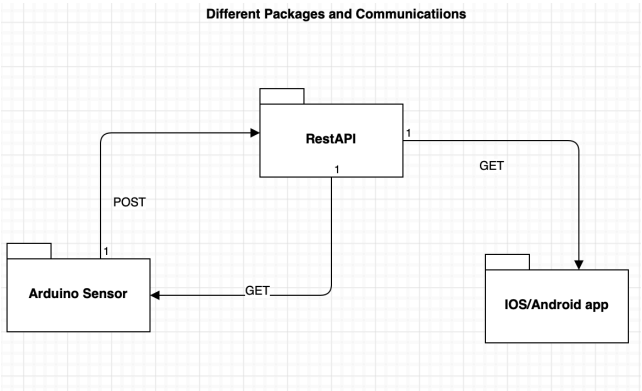


Figure2. Different objects for the project.

The following diagram shows the layout of the REST API. The model stores data, the view controls the data serialized and updates the stores data in the model. The serializer displays JSON object as View stipulates.

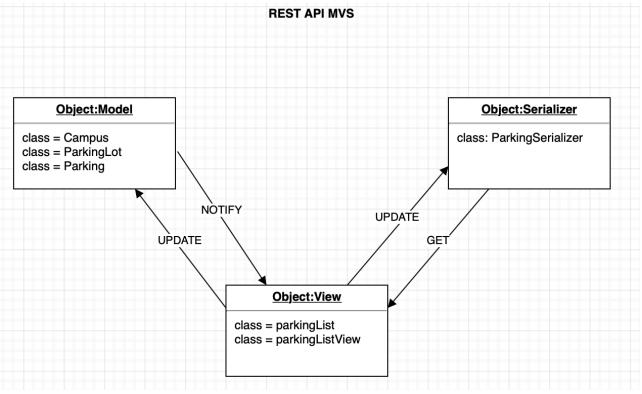


Figure3. RESTful API

The following diagram represents the structure of the mobile app, Xamarin uses an MVC pattern similar to the REST API, the only difference is that the view is not the controller, the controller controls the view and the models.

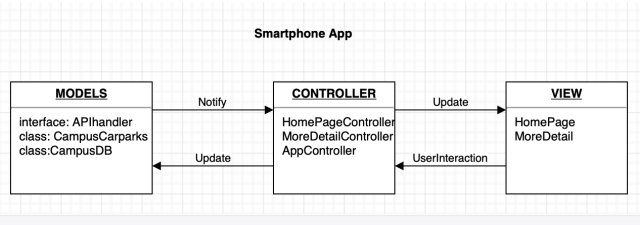


Figure 4. Structure of App

5. RESULTS

We will talk about the requirements of the application developed and the overall system. The most important part of our research is the results of the user assessments and the opinions of users on the mobile app and how it was designed.

5.1 System evaluation

The application has requirements, some of them can be measured and some cannot, these requirements ensure that the application performs optimally.
Here are some of the most important functional requirements for the system: read from the REST API server, show available parking, show a map of carparks by campus of where to find available parking spots and show a list of carparks by campus with details.
Non-functional requirements are often intangible, but as equally important. Here are the most important non-functional requirements: the app requires internet access, optimized for smartphones, the app should be able to retrieve and show available parking within a minute, the Rest API should be able to handle multiple simultaneous GET requests.

5.2 Results

5.2.1 Sample survey output

The survey had forty respondents, the results showed that most of the potential users wanted to have information about where available parking is and how to find It. More than 25% of those surveyed said that they spend more than ten minutes looking for parking on average, 81% drive to campus every weekday, 79% park for at least four hours and 41% said they prefer the North Stop carpark. The sample results from one of the questions shows that students want information, but also would prefer that there are more carparks. Some students even said that the only solution that would work is building more carparks.

5.2.2 Paper Prototype

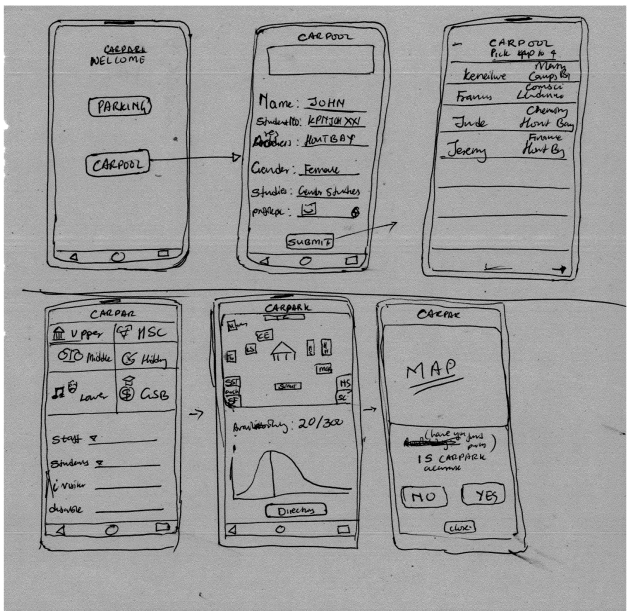


Figure 6. Paper prototype

The users were given a walkthrough of the paper prototype and given a few minutes to go through it on their own before the interview.

Through the analysis of the users' interviews most of the users were overall happy with the idea of the app, we had two paper prototypes, the first one was hated very significantly.

Users said that they prefer to click less; and they said that the paper prototype needed more clicks. The carpark codes were used to identify carparks; most of the users said that they did not know what the carpark codes meant because most of the time, they were used identifying carparks with the buildings close to them; for example, the carpark close to architecture building.

The interactions were confusing for some users, the drop-down menu for users for example, one user said that they were confused in how the drop-down was structured; they said that it should start with the most users – students (not staff) – and work its way down.

5.2.3 Interactive prototype

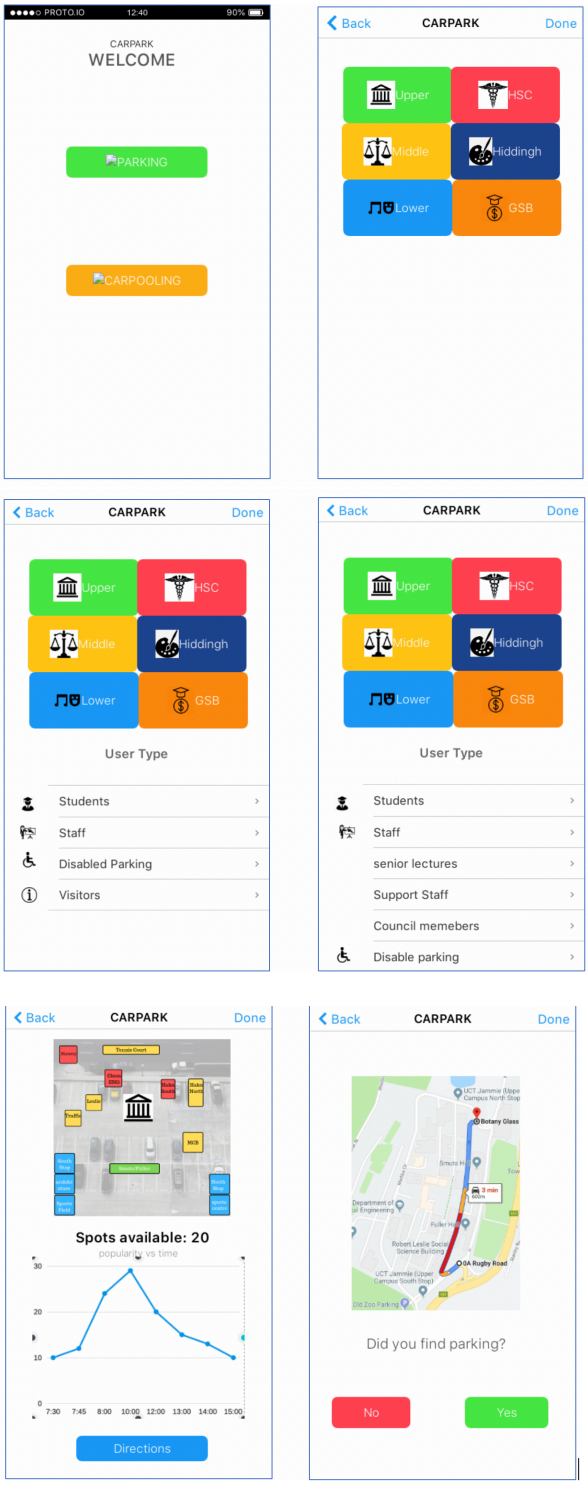


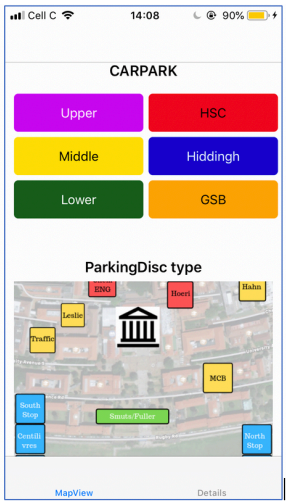
Figure 7. Interactive prototype

Screen one shows two buttons, one for carpooling and one for carparks; the carpooling feature has not yet been implemented. When a user clicks on the carpark button, they are led to the second screen which shows a grid of buttons with the campus names. When a user taps on any campus, a drop-down list with user-type shows up; and another drop down in state three if tapping on either staff or student. A user is then led to a map of the campus they chose and the carparks that they are allowed to park in; when a user taps on a carpark they get the number of spots available a distribution graph displaying utilisation. A user can finally decide whether to use the directions feature or not.

The users said that the app had too many interactions or clicks, some users said that some of the screen were unnecessary. One user said they wanted to favourite a campus and only launch the app for that campus. "The app should be simple; I feel that this app is trying to do too many things." – one user said that they only want to use the app to know about the availability of parking, and the graph has no use for them.
Most of the potential users are students, and they said that they would not find the direction feature of the app useful.

Additional features that users wanted include; being able to book a spot, virtual queueing system, users want the map to be clickable and show the carparks in a list form as well.

5.2.4 First App iteration prototype



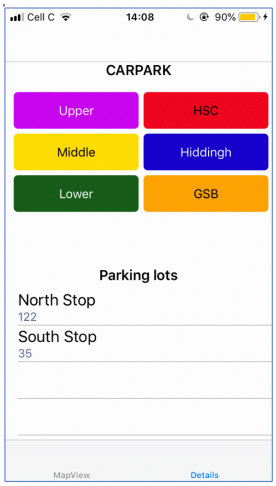


Figure 8. App iteration 1

When a user launches the app, they are greeted with screen one, a grid of the campus. When they click on a campus, a clickable map of the campus appears. A user can either click on the map or swipe left to get a detailed list of the carparks for that campus (state three). When a user clicks on the list, they are led to screen two where they see a question of whether they found parking a utility graph.

One user was involved in the paper prototype, and they said that the app is a definite improvement to the paper prototype. A user mentioned that the numbers in the list seem meaningless because they did not know whether those numbers are cars in the carpark or spots available and they did not know the capacity of said carpark as it was not in the list. The users initially did not understand the colour code of the clickable map, there was no guide or legend to help explain what the colours meant; they discussed amongst one another, until they noticed that it represented the colour of parking discs. Students and staff get to park in different places on campus, the users said that they only wanted to be shown carparks that they are allowed to park in and not all the carparks; i.e. students only want to see the blue carparks and not the red or yellow carparks.

The host asked the group what features they thought were missing from the app and would like to see or have in the next iteration. A refresh button or pulldown to refresh or an automatic refresh every certain time period, favourite a carpark (but switch out if that carpark is full), filter by parking discs and a GPS to show whether a car has parked or not instead of the button that asks users if they found parking.

5.2.4 Current Iteration

Figure 9. App iteration 2

The design for the app iteration 1 has not changed in the interactions, a user can now select a carpark as default carpark. Screen two has pull down to refresh, and that graph shows weekly data, usually data from the previous weekday. Each item in the list is colour coded by three colours: red for when there is less than 10% of parking spots available, orange when there is less than 50% but greater than 10% of parking spots and when there is greater than 50%, it is green. The colour codes are also present as the border colour of the label that shows available data in screen 3.

5.3 Discussion

5.3.1 Limitations

The time allocated for this projected meant that we were not able conduct more assessments and more experiments. Qualitative research has some limitations because of the large amount of data that has to be transcribed and analysed, and also means that we have to limit our user groups[23, 33-35].

Cross platform limitations

The application developed in this project was developed using Xamarin development framework and Xamarin forms, this was done because there seems to be an equally number of Android and iOS devices on the campuses. Xamarin forms limits users in functionality, there are certain functionality that require native app development.

5.3.2 Discussions

We have described the parking experiences of UCT students, mostly, and the problems that they often face when looking for parking. The solution we proposed was to build a low-cost, implementable smart parking system. The smart parking system uses an Arduino sensor to scan cars going in and out of carparks, to provide information so that users can use that information to find available parking spots in a carpark.

The focus of this paper was on the design and development of the mobile application following user centred design approach for the design, paired with agile development for development of the app. The prototypes were assessed by with interviews, observations, surveys and focus groups; users who regularly park on campus said that this would be a helpful app for them.

6. CONCLUSIONS

The purpose of the application is to help users find parking by giving users prior information about the availability of parking in real-time. The popular carpark that users often chose to park on at UCT was the carpark at the north end of the campus, that carpark is closer to the library, lecture theatres and the food court. In the user assessments, when the students wanted to be able to favourite a carpark, they were referring to that specific carpark.

The application provides more information, the campus map gives students carpark information relative to buildings for them to know if the closest carpark to their nearest building has spots available.

User centred design is a good design methodology that could be incorporated in a lot of projects[5, 28]. We assessed users’ needs, wants and their tasks, and created a solution that was tailored to their specific needs and wants and has functionality suiting their tasks.

UCD requires more data and more experiments to reach a good conclusion, if you pair UCD with agile, it has the potential to delay your project[26, 27, 38]. If you are solving a problem that designers and users have identified, UCD is a great design methodology; but if you are solving a problem for innovation

where a new product is the outcome, UCD may not be a great design choice. UCD can produce imitative innovation, users are often biased because of the objects they already use, thus the objects they would want are very similar to the objects they have

The assessments were conducted with less users because we had assumed that because the app is potentially solving every parking student's problems, they would want to come, without incentives. Recruiting more users, conducting more assessments, creating personas would also aid in designing. A contextual inquiry would have given us more information about the user and their experience in an actual environment, instead of a lab setting that the focus group was in. There are more qualitative assessments that we could have used for this project, they would have given a clearer picture of the results and discussions.
Xamarin mobile development is a good framework for creating a cross-platform minimal viable product, but in order to create an application with better functionality, native development either in Xamarin, Android studio or Xcode works better.

FUTURE WORK

More research in how UCD can work in agile environments, because UCD is a good design process for user facing products, such as mobile applications. UCD should be used in most user facing products, unless the designer is creating a new product segment altogether – UCD might impede in innovation.
Research in the business potential of the application; most staff and students we interacted with liked the project and therefore a good business potential.
UCD is not new, but using UCD in software engineering is not as common as it should be due to a number of reasons, but there is research about how to pair UCD with modern software engineering methodologies[8, 37, 38]. UCD when applied incorrectly or at the wrong time may cause overfitting or stymie innovation[5, 7, 8].

Carpooling a great alternative to reducing parking congestion. Using UCD to add carpooling as a feature in the near future, students are open to leaving their cars at home for at least one day and thus a good carpool product would be one that addresses users needs as well. Other universities may be able to use the framework for their campuses, even if their campuses are vastly different from UCT, the project is modular enough that a university can pick and choose, and the code is easily maintainable and modular.

Students were not happy with how parking discs are issued, and they suggested a few solutions. Time-bound parking discs, parking discs that expire after a month, three months or six months; limited weekly parking minutes, a user is a given a limit of how long they can park on campus on a weekday.
The problem of parking affects many users, so solving this problem requires user-centred design.

ACKNOWLEDGEMENTS

Joshua Benjamin worked a collaborator for this project, his paper focuses on the Arduino sensor.
We would like to thank Associate Professor Michelle Kuttel, our supervisor, she steered us in the right direction and this project might not have been a success without her leadership.

7. REFERENCES

[1] M Cardei, I Cardei, I Zankina, D Raviv. 2013. Campus Assistant Application on an Android Platform. 2013 Proceedings of IEEE Southeastcon (Jacksonville, FL, 2013), 1-6. DOI: 10.1109/SECON.2013.6567518
[2] T Kisser and S Zecha. 2017. Creating Effective Mobile Apps for National Geoparks. An Explorative Study. GI_Forum. 1, (2017), 113-125. DOI:10.1553/giscience_2017_02s113
[3] Posti, M. Schoning J, Hakkila J. 2014. Unexpected Journeys with the HOBBIT – The Design and Evaluation of an Asocial Hiking App. Proceedings of the 2014 conference on Designing interactive systems (Vancouver, 2019), 637-646. Doi:10.1145/2598510.2598592
[4] Khattak, A. and Polak, J. 1993. Effect of parking information on travelers' knowledge and behavior. Transportation. 20, 4 (1993), 373-393.
[5] Redström, J. 2006. Towards user design? On the shift from object to user as the subject of design. Design Studies. 27, 2 (2006), 123-139.
[6] Trista Lin, Hervé Rivano, Frédéric Le Mouël. A Survey of Smart Parking Solutions. IEEE Transactions on Intelligent Transportation Systems, IEEE, 2017, 18 (12), pp. 3229-3253.DOI: 10.1109/TITS.2017.2685143. hal-01501556
[7] Steen, M. Kuijt-Evers, L. Klok, J. 2007. Early user involvement in research and design projects - A review of methods and practices. 23rd EGOS Colloquium (European Group for Organizational Studies) (Vienna, 2007).
[8] McInerney, P. and Maurer, F. 2005. UCD in agile projects. interactions. 12, 6 (2005), 19-23.
[9] Mahmud, F. and Aris, H. 2015. State of Mobile Crowdsourcing Applications: A Review. 2015 4th International Conference on Software Engineering and Computer Systems (ICSECS) (kuantan, Pahang, 2015), 27-32.
[10] Fraifer, M. and Fernström, M. 2017. Designing a Smart Car Parking System (PoC) Prototype Utilizing CCTV Nodes: A vision of an IoT parking system via UCD process. Advances in Science, Technology and Engineering Systems Journal. 2, 3 (2017), 755-764.
[11] Fraifer M, Hasenfuss H, Ryan A, Elgenaidi W and Elmangoush A. 2018. It Takes Two to Tango: Merging Science and Creativity to Support Continued Innovation in the IoT Domain. Advances in Science, Technology and Engineering Systems Journal. 3, 5 (2018), 82-91.
[12] Kopecky, J. and Domingue, J. 2012. ParkJam: crowdsourcing parking availability information with linked data. The Semantic Web: ESWC 2012 Satellite Events (Heraklion, 2012), 381-386.
[13] Grazioli, A. et al. 2013. Collaborative Mobile Application and Advanced Services for Smart Parking. 2013 IEEE 14th International Conference on Mobile Data Management (Milan, 2013), 39-44.
[14] Khanna, A. and Anand, R. 2016. IoT based smart parking system. 2016 International Conference on Internet of Things and Applications (IOTA). (2016), 266-270.
[15] L Sullivan (1986). The tall office building artistically considered. Lippincott's monthly magazine. 403-409
[16]. Usability.gov,. Usability Testing. http://www.usability.gov/how-to-andtools/methods/usabilitytesting.html.
[17]. Usability.gov,. Planning a Usability Test. http://www.usability.gov/how-to-andtools/methods/planningusability-testing.html.
[18] Hackos, J and Redish, J (1998) User and task analysis for interface design. John Wiley and Sons Inc., New York, USA
[19] Hermes D. (2015) Mobile Development Using Xamarin. In: Xamarin Mobile Application Development. Apress, Berkeley, CA. DOI: https://doi.org/10.1007/978-1-4842-0214-2_1
[20] Balikas, G., Partalas, I., Kosmopoulos, A., Petridis, S., Malakasiotis, P., Pavlopoulos, I., ... & Gallinari, P. (2013).

Evaluation framework specifications. Project deliverable D, 4. http://bioasq.org/sites/default/files/PublicDocuments/BioASQ_D4.1-EvaluationFrameworkSpecification_final.pdf
[21] restfulapi.net, Creating timeless REST API. https://restfulapi.net/rest-api-design-tutorial-with-example/
[22] Fontana, A. and Frey, J.H., 2000, "The interview: from structured questions to negotiated text", in Denzin, N.K. and Lincoln, Y.S. (Eds), Handbook of Qualitative Research, 2nd ed., Sage Publications, Thousand Oaks, CA
[23] http://www.qualitative-research.net, Doing qualitative observations. http://www.qualitativeresearch.net/index.php/fqs/article/view/466/996
[24] students.uct.ac.za. Campus parking. http://www.students.uct.ac.za/students/services/transportparking/parking
[25] Laurisha Rampersad, Sarah Blyth, Ed Elson, and Michelle M. Kuttel. 2017. Improving the usability of scientific software with participatory design: a new interface design for radio astronomy visualisation software. In Proceedings of the South African Institute of Computer Scientists and Information Technologists (SAICSIT '17). ACM, New York, NY, USA, Article 29, 9 pages. DOI: https://doi.org/10.1145/3129416.3129899
[26] Sohaib O, Khan K. 2010. Integrating usability engineering and agile software development: A literature review. 2010 International Conference on Computer Design and Applications. Quinhuangdao, China. V2.32 – V2.38.
[27] Kujala Sari. 2003. User involvement: A review of benefits and challenges. Behaviour & Information Technology. 01 January 2003. 1 – 16.
[28]. http://www.ruthtrumpold.id.au, User centred design. http://www.ruthtrumpold.id.au/destech/?page_id=1486
[29] Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future generation computer systems, 29(7), 1645-1660.
[30] Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. Computer networks, 54(15), 2787-2805.
[31] Rinne, M., Törmä, S., & Kratinov, D. (2014, June). Mobile crowdsensing of parking space using geofencing and activity recognition. In 10th ITS European Congress, Helsinki, Finland (pp. 16-19).
[32] mulesoft.com, what is Rest API. https://www.mulesoft.com/resources/api/what-is-rest-api-design
[33] Preece, J., Rogers, Y., and Sharp, H. Interaction Design. Beyond Human-Computer Interaction. John Wiley & Sons, Ltd., 2002.
[34] questionpro.com, Qualitative user tests. https://www.questionpro.com/blog/qualitative-research-methods/#Qualitative_Research_Methods_with_Examples
[35] business.critizr.com. Qualitative study https://business.critizr.com/en/blog/qualitative-study-everything-you-need-to-know-about-one-on-one-interviews
[36] Alliance, Agile. "Agile manifesto." Online at http://www.agilemanifesto.org 6.1 (2001).
[37] Rettig, Marc. "Prototyping for tiny fingers." Communications of the ACM 37.4 (1994): 21-27.
[38] Kangas, Eeva, and Timo Kinnunen. "Applying user-centered design to mobile application development." Communications of the ACM 48.7 (2005): 55-59.
[39] Belitsoft.com. Xamarin vs Ionic vs React native. https://belitsoft.com/react-native-development/react-native-vs-xamarin-vs-ionic
[40] Clark J, Blake E. 2018. Advanced Software Design: Project management. CSC3003S. lecture notes. Computer Science 3003S. University of Cape Town. July-August 2018.

8. APPENDIX

A1. Survey questions

Survey questions

1.	Do you park on campus during the week?
2.	How do you go about finding parking on campus?
3.	How long does it take you to find parking?
4.	What are your preferred parking lots on campus, and why?
5.	Do you drive to campus every day of the week
6.	From what time do you usually park on campus? (e.g. 8am, 12pm or 1:30pm please don't write "around" or "between" (even though they are estimates, write numbers only please))
7.	How long do you park on campus on any given day?
8.	Do you ever experience problems looking for parking
9.	Do you experience problems trying to get out of parking?
10.	Are there any potential improvements to how parking works on campus, that you'd like to see?
11.	Have you ever carpooled?
12.	Would you be open to leaving your car at home and riding with a friend or neighbour at least once a week?
13.	Have you ever utilised any form of parking apps?
14.	what has the experience of using parking apps (if you have utilised them before)been like?
15.	If you were to help in designing a parking application, what are the features that you'd like it to have

A2. Paper prototype interview script

Paper prototype interviews.

1.	Is the app potentially accomplishing its intended goals?
2.	Did you find the application intuitive to use?
3.	Are there additional features that you think are missing that you would like to see in future iterations
4.	Novel Features that interesting to have, but not necessary?
5.	What problems did you have with the application as it is?
6.	Anything that wasn’t mentioned that you would like to add?

A3. Interactive prototype script questions

Interactive prototype.

1.	Did the application work?
2.	What were the goals of the application?
3.	Was the application intuitive to use?
4.	Did you find any problems with the use of colour?
5.	Did you find any problems with the use of icons
6.	How did you find the interactions of icons, buttons, colours and gestures in the application?
7.	Were there features that you thought were not needed for the application
8.	Which features would you like to see in the next update
9.	Are there questions that you’d like to mention, that were excluded from this survey

Having to see which parking lots have available parking. To be able to join a queue system for different parking lots so when someone leaves the first person in the queue is informed and can go to the parking. To be able to reserve a parking spot.
Show capacity for each parking area reservation system?
Reserve a parking spot/Guaranteed parking. Monitor parking availability on campus.
Show the number of spots available in different areas
How full a parking lot is
Available parking count for the different lots
Shows available bays and their type (red, yellow etc.)
Type of disk required for parking bays on the map Parking availability.
Open parking spot tracker
Knowing which parking lots are available. which one is the closest to my lecture venue
Create a platform to let people share a car together when coming to campus. It would be nice to have a sensor or some sort to see where to find empty parking spots, or count how many cars has entered the parking area. calculate the probability of finding a parking in a specific area. Allow drivers to leave a message somewhere when they are leaving which an open parking will be available. Have a parking area at Jammie stops outside UCT? What would really decrease the cars on campus, is to charge parking fee by amount of time parked instead of a whole year parking fee, use the app to pay or something
Reminder of where you parked
To find where a free parking spot is
To know where there’s available parking spots
Sensors in the bays that are activated when a car parks there or leaves
Real time updates of what is available
Carpool service to connect with people in your area
Which parking lots have bays open
A feature that can tell you where some parking is would be nice (I don’t know if that’s possible though)
For it to show which carparks have available parking
Directions to free parking
Which parking lots are open, how many spots available in each lot
When someone parks it says the lot is taken
Available parking spots

A4. Focus Group Prodding questions

Focus group

1.	How did you find the application overall in terms of the assumed goals of the design language.
2.	How did you find the interactions (pressing buttons, swiping and the gestures)?
3.	What features do you think are missing that you would want to use in the future?
4.	Which of the features in the app currently, did you regard as useless or gimmicks
5.	How do you plan to use the application? To park at the same carpark, a new carpark or change how you travel to campus based on times and availability of carparks?

A5. A sample of answers to survey question 15.